Freescale Semiconductor
应用笔记

Document Number: AN5088
Rev 0, 3/2015

Kinetis L 系列功耗管理

如何使用 Kinetis L 系列低功耗模式

内容

1 简介

Kinetis L 微控制器系列为对功耗敏感型市场提供超低功耗特性。在该 MCU 系列中实现了多种低功耗模式以满足这一需求。本应用笔记向用户展示了每种功耗模式的详细信息，并在 SDK 功耗管理演示中提供了用户案例示例。同时针对每种功耗模式提供了建议和技巧。

Kinetis 软件开发套件（SDK）向用户提供了强大的外设驱动、协议栈、中间件和应用示例，旨在简化和加速基于所有 Kinetis MCU 的应用开发。Kinetis SDK 是完全免费的，包含用于所有硬件抽象和外设驱动软件的完整源代码（在宽松的开源许可环境下）。

本应用笔记重点介绍了功耗管理控制器（PMC）、系统模式控制器（SMC）、多用途时钟产生器（MCG）和低漏电唤醒单元（LLWU）。

2 功耗模式

ARM® Cortex-M0+的功耗模式分为运行（RUN）、睡眠（SLEEP）和深度睡眠（DEEP SLEEP）。在 Kinetis L 系列中，内核使用中断唤醒指令（WFI）来启用睡眠和深度睡眠模式。该表显示了 Kinetis L MCU 的扩展功耗模式及各模式之间的联系。

表 1. Kinetis L 系列功耗模式

ARM CM0+功耗模式	Freescale MCU 功耗模式	唤醒模块	是否复位?
RUN	RUN, VLPR	-	-
RUN	CPO	AWIC/NVIC	否
SLEEP	WAIT, VLPW	NVIC	否
DEEP SLEEP	STOP, VLPS	AWIC	否
DEEP SLEEP	PSTOP1	AWIC	否
DEEP SLEEP	PSTOP2	AWIC/NVIC	否
DEEP SLEEP	LLS	LLWU	否
DEEP SLEEP	VLLS0/1/3	LLWU	LLWU 复位

NVIC 意味着任何中断源可将 MCU 从 WAIT/VLPW 模式唤醒。AWIC 意味着只有参考手册中列出的 AWIC 唤醒源才能将 MCU 从 STOP/VLPS 模式唤醒。LLWU 意味着只有参考手册中列出的 LLWU 唤醒源才能将 MCU 从 LLS/VLLSx 模式唤醒。要从 VLLSx 模式唤醒，需要经历复位流程并调用 LLWU 复位。在计算运行模式 (CPO) 下，ARM 内核处于运行模式。任何异步中断和 ARM 内核同步中断可以将 MCU 唤醒，使其进入运行模式。

对于 Kinetis L 系列器件，NMI 引脚可以唤醒所有功耗模式；如果复位引脚未被总线时钟过滤，则复位引脚可以将 MCU 的功耗模式重置为默认的运行（RUN）模式。

2.1 功耗模式详细信息

在不同的 Kinetis L 系列器件中使用的功耗模式不尽相同。用户可以查看相关的参考手册以获取详细信息。下文介绍了每种功耗模式，其中包含模式的相关详细信息以及模式进入和退出的基本信息。有关更多高级信息，包括阻止进入低功耗模式的因素，请参阅参考手册中的功耗模式转换章节。以下给出的测量数据、频率和其他极限数据仅供参考。请确保使用 MCU 数据手册中的正式值。

2.1.1 运行模式（RUN）

- 在任意复位后进入（默认情况下，FOPT = 0xFF）。
- 片内电压调节器开启，全功能。
- 堆栈指针（SP）、程序计数器（PC）和链接寄存器均被设置。
- ARM Cortex-M0+处理器退出复位并从 Flash 地址 0x0 和 0x4 处读取起始 SP 和 PC。
- 默认情况下只使能 DMA 和 Flash 时钟。
- 若不使用，通过清零 SCGCx 寄存器中的外设时钟门控位以降低功耗。

在运行模式下，室温下期望的典型 IDD 为 4 至 6 mA（查看 MCU 器件数据手册以获取更多信息），这取决于时钟频率和 MCU 系列。一些 Kinetis L 系列器件支持启动进入 VLPR 模式。查看参考手册的第 6 章以获取更多信息。

2.1.2 超低功耗运行模式（VLPR）

- 在任意复位后选择（如果 MCU 支持该特性，且 FOPT 具有 VLPR 启动设定）。
- 片内电压调节器处于仅提供 MCU 低频运行所需电压的模式。
- 内核频率限制为最高 4 MHz（应将时钟分频器设置为合适的值以满足该限制）。
- 总线/Flash 频率限制为最高 800 kHz–1 MHz（总线时钟与 Flash 共用时钟分频器）。
- 清零 SCGCx 寄存器中的时钟门控位以降低功耗。
- 不允许进行 Flash 编程和擦除操作。

- 只有在选择内部 IRC（4 MHz 或 8/2 MHz）时才可以从 BLPI 模式下进入 VLPR。
- 如果外部时钟低于 16 MHz，那么可从 BLPE 模式进入 VLPR。

在 VLPR 模式下，室温下期望的典型 IDD 为 0.16 至 0.5 mA（查看 MCU 器件数据手册以获取更多信息）。由于 MCG 设置的不同，内部 IRC 的最大频率也因 MCU 而异。一些 Kinetis L 系列器件还支持启动进入 VLPR 模式。用户应查看参考手册以获取更多信息。

2.1.3 待机模式（WAIT）

- NVIC 仍然对中断敏感。
- 继续为外设提供时钟。
- 若不使用，通过清零 SCGCx 寄存器中的外设时钟门控位以降低功耗。
- 中断时，ARM Cortex-M0+内核退出睡眠模式，恢复运行模式处理。

在室温下，期望的 IDD 约为 RUN IDD - 2（查看 MCU 器件数据手册以获取更多信息）。

2.1.4 超低功耗等待模式（VLPW）

- 仅可从 VLPR 进入 VLPW。
- 片内电压调节器处于仅提供 MCU 低频运行所需电压的模式。
- NVIC 仍然对中断敏感。
- 继续为外设提供时钟。
- 若不使用，通过清零 SCGCx 寄存器中的外设时钟门控位以降低功耗。
- 可选择性地保持外部参考时钟使能（最大值为 16 MHz）。
- 中断时，ARM Cortex-M0+内核退出睡眠模式，恢复 VLRP 模式处理。

在室温下，期望的 IDD 约为 VLRP IDD - 0.08 至 0.1 mA（查看 MCU 器件数据手册以获取更多信息）。

2.1.5 常规停止模式（STOP）

- 异步唤醒中断控制器（AWIC）用于从中断唤醒。
- 系统和外设时钟停止。
- 所有 SRAM 内容保留且保持 I/O 状态。
- 可从任意正常运行模式进入停止模式。
- MCG 模块可配置用于使参考时钟保持运行状态。
- 可选择性地保持 PLL 使能，但是输出会被关闭—C5[PLLSTEN]。
- 可选择性地保持内部参考时钟使能—C1[IREFSTEN]。
- 可选择性地保持外部参考时钟使能—OSC_CR[EREFSTEN]。
- 将在进入停止模式时的相同 MCG 模式下退出停止模式，除了在处于 PEE 模式时进入停止模式；并且在退出停止模式时， MCG 将处于 PBE 模式。

在室温下，期望的 IDD 约为 200 uA（查看 MCU 器件数据手册以获取更多信息）。

2.1.6 局部停止模式 1（PSTOP1）

- 异步唤醒中断控制器（AWIC）用于从中断唤醒。
- 系统和外设时钟停止。
- 保留所有 SRAM 内容且保持 I/O 状态。
- 可从任意运行模式进入 PSTOP1 模式。
- MCG 和 PMC 模块的状态不变。
- 进入 PSTOP1 后，将在相同的 MCG 模式下退出 PSTOP1 模式。

室温下，期望的 IDD 约为 2 mA，具体取决于内核速度和启用的外设。

2.1.7 局部停止模式 2（PSTOP2）

- AWIC/NVIC 用于从中断唤醒。
- 系统时钟停止，继续为外设提供时钟。
- 保留所有 SRAM 内容且保持 I/O 状态。
- 可从任意运行模式进入 PSTOP2 模式。
- MCG 和 PMC 模块的状态不变。
- 进入 PSTOP2 后，将在相同的 MCG 模式下退出 PSTOP2 模式。

室温下，期望的 IDD 约为 2 mA，具体取决于内核速度和启用的外设。

2.1.8 计算运行模式（CPO）

- AWIC/NVIC 用于从中断唤醒。
- 使能对于 MCU 内核和 SRAM/Flash 的访问，停止其他外设时钟。
- 可从任意运行模式进入 CPO 模式。
- MCG 和 PMC 模块的状态不变。

在室温下，期望的 IDD 几乎与运行模式相同。

2.1.9 超低功耗停止模式（VLPS）

- NVIC 禁用。
- AWIC 用于从中断唤醒。
- 系统和外设时钟停止。
- 所有 SRAM 保留内容且保持 I/O 状态。
- 可从任意 MCG 模式进入 VLPS 模式。
- MCG 模块可配置用于使参考时钟保持运行状态。
- 可选择性地保持内部参考时钟使能—C1[IREFSTEN]。
- 可选择性地保持外部参考时钟使能—OSC_CR[EREFSTEN]。
- 将在进入 VLPS 模式时的相同 MCG 模式下退出 VLPS 模式，除了在处于 PEE 模式时进入 VLPS 模式；并且在退出 VLPS 模式时， MCG 将处于 PBE 模式。

在室温下，期望的 IDD 约为 2 uA（查看 MCU 器件数据手册以获取更多信息）。

2.1.10 低漏电停止模式（LLS）

- NVIC 禁用。
- AWIC 禁用。
- 配置 LLWU 以使能所需的唤醒源。
- 系统和外设时钟停止。
- 所有 SRAM 内容保留且保持 I/O 状态。
- 大多数外设都处于状态保持模式。
- 可从任意 MCG 模式进入 LLS 模式。
- 在低范围和低功耗振荡器模式（32 kHz）下，可选择性保持外部参考时钟使能。
- MCG 为静态，无时钟激活（IREFSTEN 和 PLLSTEN 无影响）。
- 将在进入 LLS 模式时的相同 MCG 模式下退出 LLS 模式，除了在处于 PEE 模式时进入 LLS 模式；并且在退出 LLS 模式时， MCG 将处于 PBE 模式。

- 发生唤醒事件时，SRS 寄存器中的 WAKEUP 位将置位。
- 执行 LLWU 中断代码后，会继续执行 LLS 模式进入指令的下一条指令。

在室温下，期望的 IDD 约为 2 uA（查看 MCU 器件数据手册以获取更多信息）。并非所有的 Kinetis MCU L 器件均可实现该功耗模式。详情参见参考手册。

2.1.11 极低漏电停止 3 模式（VLLS3）

- NVIC 禁用。
- AWIC 禁用。
- 配置 LLWU 以使能所需的唤醒源。
- 系统和外设时钟停止。
- 大多数模块禁用。
- 保留所有 SRAM 内容且保持 I/O 状态。
- 可从任意 MCG 模式进入 VLLS3 模式
- 在低范围和低功耗振荡器模式（32 kHz）下，可选择性保持外部参考时钟使能。
- MCG 关闭，无时钟激活（IREFSTEN 和 PLLSTEN 不起作用）。
- 发生唤醒事件时，将通过复位流程退出 VLLS3。MCG 处于 FEI 模式。
- 发生唤醒事件时，SRS 寄存器中的 WAKEUP 位将置位，并且 MCU 执行来自复位向量的代码。然后，当 LLWU 中断向量使能时，执行 LLWU NVIC 中断。

在室温下，期望的 IDD 约为 1 uA（查看 MCU 器件数据手册以获取更多信息）。该功耗模式在所有最新的 Kinetis MCU L 器件上均可实现。详情参见参考手册。

2.1.12 极低漏电停止 1 模式（VLLS1）

- NVIC 禁用。
- AWIC 禁用。
- 配置 LLWU 以使能所需的唤醒源。
- 系统和外设时钟停止。
- 大多数模块禁用。
- 不保留 SRAM 内容，保持寄存器文件和 I/O 状态。
- 可从任意 MCG 模式进入 VLLS1 模式。
- 在低范围和低功耗振荡器模式（32 kHz）下，可选择性保持外部参考时钟使能。
- MCG 关闭，无时钟激活（IREFSTEN 和 PLLSTEN 不起作用）。
- 发生唤醒事件时，将通过复位流程退出 VLLS1。MCG 处于 FEI 模式。
- 发生唤醒事件时，SRS 寄存器中的 WAKEUP 位将置位，并且 MCU 执行来自复位向量的代码。然后，当 LLWU 中断向量使能时，执行 LLWU NVIC 中断。

在室温下，期望的 IDD 约为 0.5 uA（查看 MCU 器件数据手册以获取更多信息）。该功耗模式在所有最新的 Kinetis MCU L 器件上均可实现。详情参见参考手册。

2.1.13 极低漏电停止 0 模式（VLLS0）

- NVIC 禁用。
- AWIC 禁用。
- 配置 LLWU 以使能所需的唤醒源。
- 系统和外设时钟停止。
- 可选择性地使能上电保护（POR）。
- 不保留 SRAM 内容，保持寄存器文件和 I/O 状态。
- 可从任意 MCG 模式进入 VLLS0 模式。
- 可选择性地关闭低功耗振荡器（1 KHz）时钟。

- 在低范围和低功耗振荡器模式（32 kHz）下，可选择性保持外部参考时钟使能。
- MCG 关闭，无时钟激活（IREFSTEN 和 PLLSTEN 不起作用）。
- 发生唤醒事件时，将通过复位流程退出 VLLS0。MCG 处于 FEI 模式。
- 发生唤醒事件时，SRS 寄存器中的 WAKEUP 位将置位，并且 MCU 执行来自复位向量的代码。当 LLWU 中断向量使能时，执行 LLWU NVIC 中断。

在室温下，期望的 IDD 约为 0.3 uA（查看 MCU 器件数据手册以获取更多信息）。该功耗模式在所有的 Kinetis MCU L 器件上均可实现。详情参见参考手册。

2.2 功耗模式转换

本章节描述了用于 Kinetis L 系列的功耗模式转换。要进入 VLPx/LLS/VLLSx 模式，用户必须通过在 SMC_PMPROT 中设置相关位来使能模式进入。该寄存器为一次写入，并且由任意非 VLLSx 模式恢复的复位来复位。在尝试进入这些低功耗模式之前，用户必须先在寄存器中使能所有功耗模式进入。一些最新的 Kinetis MCU 允许 VLPR 启动模式，这需要将 Flash 配置字段 0x40D 中的 bit 4 设置为 0（0bxxx0xxxx）。在框图中以虚线表示。请注意，任意复位指的是该 MCU 的所有复位类型，因此当运行 VLLSx 模式恢复时，可能进入 VLPR 模式（如果使能）。

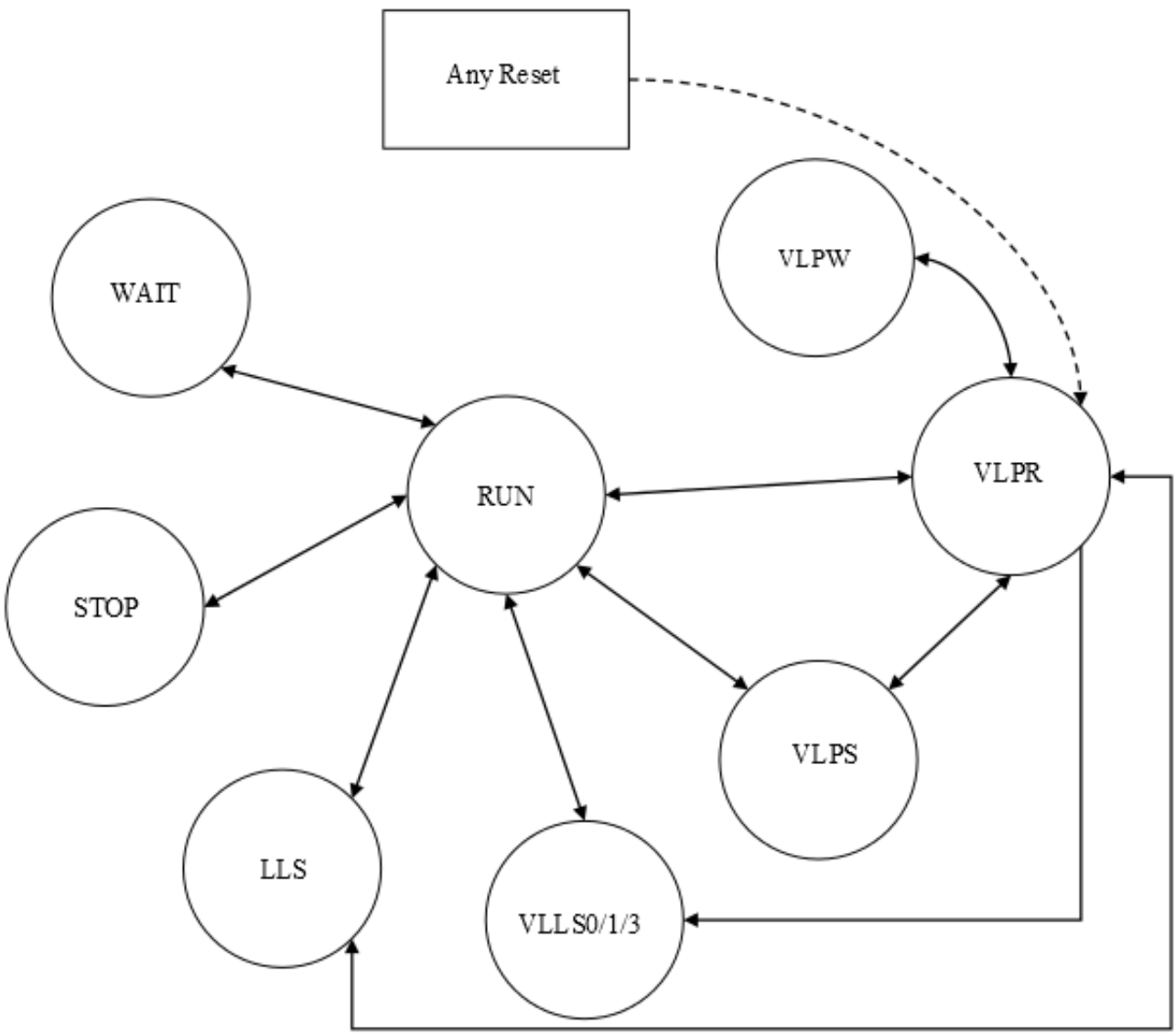

图 1. 功耗模式转换

3 MCG 和 MCG_Lite

在 Kinetis L 系列中使用两种类型的 MCG。一个是基于 FLL/PLL 的 MCG，另一个是基于 IRC 的 MCG，称为 MCG_Lite。表中给出了 MCG 和 MCG_Lite 的差别。

表 2. MCG 和 MCG_Lite

特性	MCG	MCG_Lite
高频源	FLL 或 PLL——可配置	HIRC——固定为 48 MHz
低频源	4 MHz 和 32 kHz IRC	8/2 MHz IRC
自动调节	是（4 MHz 和 32 kHz IRC）	否（出厂已调）
外部时钟监视器	是	否
VLPR 时钟源	4 MHz IRC/Ext OSC	8/2 MHz IRC/Ext OSC
STOP/LLS 恢复 MCG 模式	无变化，除了 PEE（返回 PBE 模式）	无变化，返回相同的 MCG 模式
复位后默认 MCG 模式	FEI 模式	8 MHz IRC

用户应参见参考手册，了解具体 MCU 器件中的 MCG 用法。在所有器件中，MCG 的设置可能会不同。

4 快速入门指南

本快速入门指南基于 SDK 1.1 GA 功耗管理演示程序（power_manager_demo）。在进入某个功耗模式前，用户需要初始化和注册所有的功耗参数结构和唤醒源。在本演示程序中，采用 RTC 作为唤醒源。以下是示例代码。

初始化参数和唤醒源

```
// Power mode protection initialization
// Must make sure all request power mode entry are allowed
#ifdef SYSTEM_SMC_PMPROT_VALUE
  SMC->PMPROT = SYSTEM_SMC_PMPROT_VALUE;
#endif

//define a power manager user config list for all power mode will use
    power_manager_user_config_t vlpwConfig;
#if FSL_FEATURE_SMC_HAS_STOP_SUBMODE0 & BOARD_SW_HAS_LLWU_PIN
    power_manager_user_config_t vlls0Config;
#endif
    power_manager_user_config_t vlls1Config;
#if FSL_FEATURE_SMC_HAS_STOP_SUBMODE2
    power_manager_user_config_t vlls2Config;
#endif
    power_manager_user_config_t vlls3Config;
#if FSL_FEATURE_SMC_HAS_LOW_LEAKAGE_STOP_MODE
    power_manager_user_config_t llsConfig;
#endif
    power_manager_user_config_t vlpsConfig;
    power_manager_user_config_t waitConfig;
    power_manager_user_config_t stopConfig;
    power_manager_user_config_t runConfig;
#if FSL_FEATURE_SMC_HAS_HIGH_SPEED_RUN_MODE
    power_manager_user_config_t hsrunConfig;
#endif
// Example of constant configuration
// It may save the space in RAM
const power_manager_user_config_t vlprConfig = {
```

```
        .mode = kPowerManagerVlpr,
#if FSL_FEATURE_SMC_HAS_LPWUI
        .lowPowerWakeUpOnInterruptOption = true,
        .lowPowerWakeUpOnInterruptValue = kSmcLpwuiEnabled,
#endif
        .sleepOnExitValue = false,
        .sleepOnExitOption = false
};
// Initializes array of pointers to power manager configurations
power_manager_user_config_t const *powerConfigs[] =
{
   &vlls0Config,
   &llsConfig,
&stopConfig,
   &runConfig,
   &vlprConfig
};

// User callback data
user_callback_data_t callbackData0;
// Initializes callback configuration structure for power manager
const power_manager_callback_user_config_t callbackCfg0 = { callback0,
        kPowerManagerCallbackBeforeAfter,
        (power_manager_callback_data_t*) &callbackData0 };

// Initializes array of pointers to power manager callbacks
power_manager_callback_user_config_t const * callbacks[] =
{ &callbackCfg0 };

// Initializes hardware
hardware_init();
// Initializes OS abstraction layer which uses LPTMR HAL layer
OSA_Init();

// Make the current Clock Manager mode configuration 1 (default configuration)
/* Set clock configurations to clock manager. */
CLOCK_SYS_Init(g_defaultClockConfigurations, CLOCK_CONFIG_NUM,
                  &clockCallbackTable, ARRAY_SIZE(clockCallbackTable));

CLOCK_SYS_UpdateConfiguration(cmConfigMode, kClockManagerPolicyForcible);

// Using RTC as interrupt source for this demo
#ifndef FRDM_K22F120M
    // Configure RTC pins
    configure_rtc_pins(BOARD_RTC_FUNC_INSTANCE);
#endif
    // select the 1Hz for RTC_CLKOUT
    CLOCK_SYS_SetRtcOutSrc(kClockRtcoutSrc1Hz);

RTC_DRV_Init(0);

// Set a start date time and start RTC
date.year = 2014U;
date.month = 4U;
date.day = 30U;
date.hour = 14U;
date.minute = 0U;
date.second = 0U;
RTC_DRV_SetDatetime(0, &date);

// Initializes debug UART console
dbg_uart_init();

// Initializes GPIO driver for LEDs and buttons
GPIO_DRV_Init(switchPins, ledPins);

memset(&callbackData0, 0, sizeof(user_callback_data_t));

// initializes configuration structures
```

```
vlpwConfig = vlprConfig;
vlpwConfig.mode = kPowerManagerVlpw;

// VLLS0 mode is supported only by some SOCs.
vlls0Config = vlprConfig;
vlls0Config.mode = kPowerManagerVlls0;

// classic LLS mode retains all ram content too
llsConfig = vlprConfig;
llsConfig.mode = kPowerManagerLls;

stopConfig = vlprConfig;
stopConfig.mode = kPowerManagerStop;

runConfig.mode = kPowerManagerRun;

// initialize power manager driver
POWER_SYS_Init(&powerConfigs,
sizeof(powerConfigs)/sizeof(power_manager_user_config_t *),
    &callbacks,
    sizeof(callbacks)/sizeof(power_manager_callback_user_config_t *));

// Enables LLWU interrupt
INT_SYS_EnableIRQ(LLW_IRQn);
```

4.1 VPLR 模式的进入和退出示例

4.1.1 针对 VLPR/VLPW 的 MCG/MCG_Lite 模式变换

要进入 VLPR/VLPW 模式，用户需要将 MCG 模式切换到 BLPI/BLPE 模式，或切换 MCG_Lite 到 8/2 MHz IRC 或 EXT 模式。SDK 时钟驱动程序 CLOCK_SYS_UpdateConfiguration 提供了这种功能。用户示例代码和 API 如下所示：

MCG 模式变换示例

```
//If apps default CM config mode is not VLPR, but needs to enter VLPR, and real CM
config
 //is not VLPR, then we need to update it to VLPR mode here. Otherwise pass through
if ((cmConfigMode != CLOCK_CONFIG_INDEX_FOR_VLPR) &&
(CLOCK_SYS_GetCurrentConfiguration() != CLOCK_CONFIG_INDEX_FOR_VLPR))
{
     CLOCK_SYS_UpdateConfiguration(CLOCK_CONFIG_INDEX_FOR_VLPR, kClockManagerPolicyForcible);
}
```

关于如何使用这些 API 的更多详情，请参见 API 用户指南。

4.1.2 VLPR 模式进入

SDK 驱动程序 POWER_SYS_SetMode 为用户提供了进入特定功耗模式的方法。示例代码与下列类似：

VLPR 模式进入示例

```
mode = kDemoVlpr;
//If apps default CM config mode is not VLPR, but needs to enter VLPR, and real CM config
is not VLPR, then we need to update it to VLPR mode here. Otherwise pass through.
 if ((cmConfigMode != CLOCK_CONFIG_INDEX_FOR_VLPR) &&
(CLOCK_SYS_GetCurrentConfiguration() != CLOCK_CONFIG_INDEX_FOR_VLPR))
{
 CLOCK_SYS_UpdateConfiguration(CLOCK_CONFIG_INDEX_FOR_VLPR, kClockManagerPolicyForcible);
```

```
}
ret = POWER_SYS_SetMode(mode, kPowerManagerPolicyAgreement);
```

关于如何使用这些 API 的更多详情，请参见 API 用户指南。

4.1.3 VLPR 模式退出

Kinetis L 系列器件不支持中断返回正常运行模式，因此用户需要使用功耗管理功能返回正常运行模式。

VLPR 模式退出示例

```
mode = kDemoRun;
ret = POWER_SYS_SetMode(mode, kPowerManagerPolicyAgreement);
// update Clock Mode if user want to go back to normal speed mode
updateClockManagerToRunMode(cmConfigMode);
```

4.2 LLS 模式的进入和退出示例

将模式值设为 kDemoLls。调用 POWER_SYS_SetMode 以进入 LLS 模式，此时 RTC 用作 LLWU 唤醒源（已配置）。

LLS 模式的进入和退出示例

```
mode = kDemoLls;
ret = POWER_SYS_SetMode(mode, kPowerManagerPolicyAgreement);
// update Clock Mode if user want to go back to normal speed mode
updateClockManagerToRunMode(cmConfigMode);
```

4.3 VLLS0 模式的进入和退出示例

将模式值设为 kDemoVlls0。调用 POWER_SYS_SetMode 以进入 VLLS0 模式，此时 RTC 用作 LLWU 唤醒源（已配置）。

VLLS0 模式的进入和退出示例

```
mode = kDemoVlls0;
ret = POWER_SYS_SetMode(mode, kPowerManagerPolicyAgreement);
```

5 较低功耗模式的使用提示

用户应检查 Kinetis L 系列器件中的以下注意事项。其中某些事项可应用于其他 Kinetis 系列 MCU。

1. 使用 NMI 引脚作为 VLLSx 模式的开关或唤醒源。当使用该引脚时，VLLSx 恢复将经历一个复位流程。在 MCU 复位后，如果 NMI 引脚仍为有效，MCU 将进入 NMI 中断并且有可能访问其他外设寄存器。同时，禁用其他所有外设。这会导致硬件故障中断，并且可能引起 WDOG 复位。在这种情况下，用户将需要通过一个合适的 NMI 中断例程来处理该问题。例如，当从 VLLSx 模式恢复时，NMI 中断例程会先禁用 WDOG，并且在访问寄存器前先确保外设时钟使能。
2. 如果 NMI 引脚用于其他功能，例如用 PTA4 唤醒 MCU，则应通过清零 Flash 配置字段 0x40D 中的 bit 2 来禁用 NMI 功能。在对 Kinetis L MCU 进行首次编程时，应确保该引脚不会被拉低到 0。否则，用户对 Flash 进行编程时会失败。
3. Kinetis L MCU 采用 CMOS 电路。用户必须确保任何使用的 GPIO 输入静态电压为 0 或 VDD。否则，将导致漏电流（特别是当输入为 VDD/2 时）。
4. 保持所有未使用/未连线的引脚为已知值，如果这些引脚默认不禁用，则使其输出低电平或高电平，以获取最佳的低功耗模式性能。如果任意引脚悬空或其输入电压不为 0 或 VDD，则将导致较大的漏电流。

5. 将所有不用的外设或时钟源禁用。

6 参考资料

下列参考文献包含与 Kinetis 系列功耗管理相关的其他信息。您可以在 Kinetis (www.freescale.com/Kinetis) 网页上选择一个器件或特定的系列来查看具体的参考手册、数据手册或勘误表，以了解更多信息。如需查找最新的 SDK 安装程序，请访问 www.freescale.com/kdsdk 。如需查找以下应用笔记，请搜索文档编号。

- **MCU** 参考手册：参考手册的“芯片配置”章节包含特定 MCU 的设置详情，并对每个 MCU 的复位和功耗管理特性进行了详细说明。
- **MCU** 数据手册：数据手册包含所有 MCU 规格，包括时钟频率、低功耗模式的功耗预期值和更多内容。
- **MCU** 勘误表：器件勘误表标识因 MCU 问题而未能实现的功能和/或规格。大多数问题都有解决办法。
- ***Freescale MQX***™低功耗管理（**AN4447**）：Freescale 提供 MQX，一款全功能的免费实时操作系统（RTOS）。从版本 3.8 开始，MQX 集成了低功耗管理（LPM）驱动程序，可以在 MQX 应用程序中利用低功耗工作模式。
- 在 ***Kinetis*** 系列上使用低功耗模式（**AN4470**）：包含可在带模式控制器 2 的 Kinetis 和 ColdFire+器件上使用的低功耗模式进入演示代码。
- 用于 ***Kinetis MCU*** 的功耗管理（**AN4503**）：包含 Kinetis MCU 功耗模式架构介绍、功耗模式测量和功耗管理技术。

7 修订历史记录

表 3. 修订历史记录

修订版本号	日期	重要改动
0	02/2015	初始版本

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

本文档中的信息仅供系统和软件实施方使用 Freescale 产品。 本文并未明示或者暗示授予利用本文档信息进行设计或者加工集成电路的版权许可。 Freescale 保留对此处任何产品进行更改的权利，恕不另行通知。

Freescale 对其产品在任何特定用途方面的适用性不做任何担保、表示或保证，也不承担因为应用程序或者使用产品或电路所产生的任何责任，明确拒绝承担包括但不局限于后果性的或附带性的损害在内的所有责任。 Freescale 的数据表和/或规格中所提供的“典型”参数在不同应用中可能并且确实不同，实际性能会随时间而有所变化。 所有运行参数，包括“经典值”在内，必须经由客户的技术专家对每个客户的应用程序进行验证。 Freescale 未转让与其专利权及其他权利相关的许可。 Freescale 销售产品时遵循以下网址中包含的标准销售条款和条件：freescale.com/SalesTermsandConditions.

Document Number AN5088
Revision 0, 3/2015